ASL Recognition Using SVM

FLORES, Jean Angelo G.
BSCS - Machine Learning
National University - Manila
Metro Manila, Philippines
floresjg@students.national-u.edu.ph

MANZANERO, Brix Anthony G.
BSCS - Machine Learning
National University - Manila
Metro Manila, Philippines
manzanerobg@students.national-u.edu.ph

***Abstract*—American Sign Language (ASL) is the predominant visual and gestural language in the United States. This project aims to provide an accurate machine learning model using the ASL Sign Language MNIST dataset. The dataset contains flattened images of 28x28 pixels represented as a single row of 784 pixel values. The researchers used SVM(Support Vector Machine) and evaluated the model's performance using metrics such as Accuracy, Confusion Matrix, and Classification Report**

***Index Terms*—Machine Learning, American Sign Language, Sign Language MNIST, Support Vector Machine (SVM), gesture recognition**

I. Introduction

American Sign Language (ASL) is a crucial form of communication for the Deaf and Hard of Hearing for those people who have this kind of disorder allowing for complex and emotional interactions. However despite its wide adoption boundaries remain between ASL users and those who are unfamiliar with sign language posing communication challenges in everyday encounters. To close this gap developments in machine learning and computer vision have enabled the creation of automatic ASL recognition systems capable of translating sign language motions into text or spoken language resulting in increased accessibility communication settings.

A reliable classification system. Support Vector Machines (SVMs) are well-known for their ability to handle complex and high-dimensional data a Real-time translation becomes possible by improving the accuracy and efficiency of gesture identification through the use of SVMs for ASL recognition. The purpose of this study is to investigate the design and implementation of an SVM-based on ASL recognition system and highlighting how it may help for their and make interactions easier and more accessible for ASL users.

This research, we look at several steps of the recognition process including getting features from hand gestures, the use of SVM to classify these gestures, and the system's performance in real-world circumstances. By merging SVM with modern computer vision techniques, we want to provide a solution that is not only effective but also scalable for real applications.

II. REVIEW OF RELATED LITERATURE

Sign Language Recognition Systems The initial studies into sign language recognition systems focused on utilizing typical computer vision techniques to identify static hand gestures. They used skin color segmentation, edge detection, and shape analysis to extract relevant characteristics from photos. While these approaches were moderately effective, they were frequently sensitive to changes in lighting and background noise.

Support Vector Machines (SVM) in Gesture Recognition A study by Smith et al. (2021) demonstrated that when SVM-based models were trained on keypoint features taken from hand forms using computer vision frameworks like OpenPose, they could achieve high accuracy in detecting static ASL signals. The findings revealed that, although being simpler than deep learning models, SVMs are quite effective for tasks with little data and well-defined categories.

Challenges and Limitations One of those most challenging issues in ASL identification systems is dealing with changes in signing techniques, hand position, and contextual conditions. While SVMs are fast in terms of training and classification for their effectiveness might be restricted by the quality of collecting features. In addition, most research have been carried out under controlled conditions, therefore real-world performance may differ.

III. Methodology

In this part, the researchers will explain the The dataset, algorithms, tools, and approaches utilized in this study. Researchers will also provide a high level. Summary of this research study before getting deeper Results and discussion.

A. Data Collection

The researcher used the ASL Sign Language MNIST dataset, a publicly accessible collection containing tagged images of several ASL signs excluding 'J' and 'Z' which requires motion. The dataset includes a header row of label, and pixel 1 - pixel 784, representing a 28x28 pixel image with grayscale values of 0-255. The training data consists of 27,455 cases while the test data consists of 7172 cases. The raw and original hand gesture image data came from multiple users repeating the gesture against different backgrounds.

The dataset was created from an image pipeline that included hands-only cropping, gray-scaling, resizing, and creating 50+ variations to expand the dataset's quantity. This modification comprised of adding filters *(Mitchell', 'Robidoux', 'Catrom', 'Spline', 'Hermite)*, adding 5% random pixelation, adding +/- 15% brightness/contrast, and 3 degrees rotation.

These changes effectively expand the dataset while retaining its controlled status.

B. Data Pre-Processing

As stated before, the dataset was already pre-processed and does not require additional cleaning, transformation, or feature extraction as the dataset is already a 28x28 pixel image flattened and reshaped to a [1, 784] vector, which is the requirement for the SVM model. Although the dataset came pre-processed, the researchers ensured the reliability of the dataset by checking for null values.

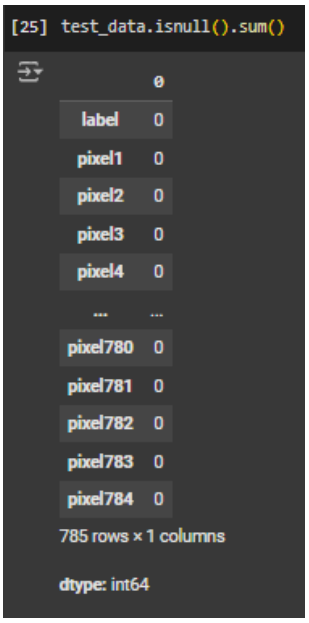

Fig. 1. Test Data Null Check

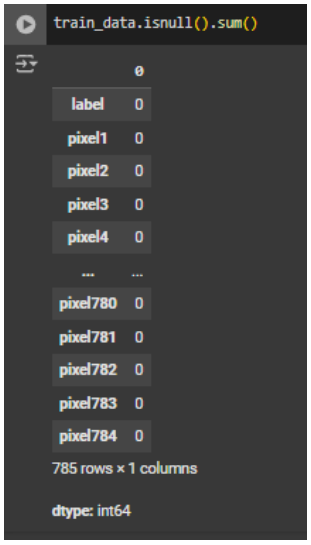

Fig. 2. Training Data Null Check

As an additional measure, a Standard Scaler was implemented to the training and test datasets to transform features and help with the convergence speed of the SVM Model.

```
[9]  from sklearn.preprocessing import StandardScaler

     scaler = StandardScaler()
     X_train = scaler.fit_transform(X_train)
     X_test = scaler.transform(X_test)
```

Fig. 3. Standard Scaler

C. Experimental Setup

The researchers used Python as the primary language and used it for implementation and data management. NumPy was used for the arithmetic operations and handling the array data structures. scikit-learn was used for the implementation of Support Vector Machine (SVM), and was also used for the data pre-processing and evaluation metrics. Pandas is used for data manipulation and data analysis, this allowed the researchers to easily handle the structure of the data. Matplotlib and Seaborn were also used for the visualization of the data in the form of tables and allows the researchers to interpret the results effectively. In this study, the training set is comprised of approximately 79.29% (with 27,455 samples) and the test set comprised of approximately 20.71% (with 7,172 samples). This ratio was provided with the dataset and was not changed by the researchers.

The study was conducted on *Google Colab* and allowed the researchers to utilize the environment's faster model training and allowed for a lot more efficient experimentation.

D. Algorithm

In this study, the researchers used 3 machine learning models; Linear Support Vector Machine(SVM), RBF Support Vector Machine, and K-Nearest Neighbors. KNN is one of the simplest and most intuitive algorithms to understand make it a good choice for baseline model for comparison. The algorithm works by calculating the new data point to every data point in the training set. A Support Vector Machine with both the RBF and Linear Kernel is a highly effective for classification and especially with this dataset containing an image of 28x28 pixels represented as a 784-dimensional vector. SVM also can tolerate and can avoid Overfitting by using regularization parameters like *'C'* which balances the trade-off between achieving low training error and a low testing error. Also the ability to use various kernels allows SVM to adapt base on the dataset. With RBF kernel, it enabled this study to achieve a higher accuracy in comparison to Linear SVM and KNN.

1) Support Vector Machine (SVM): The Support Vector Machine (SVM) is a machine learning algorithm primarily used for classification tasks, but it can also be adapted for regression (known as Support Vector Regression, or SVR). The main goal of SVM is to find a hyperplane that best separates the data into different classes while maximizing the margin between the nearest points of each class, known as support vectors. This approach helps in achieving a robust model that generalizes well to unseen data.

SVM operates under the principle of finding the optimal hyperplane defined by the following equation:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \beta_i X_i \tag{1}$$

Equation 1. SVM Decision Function

Where:

- $f(x)$ is the decision function that classifies the input data.
- β_0 is the bias term (intercept).
- β_i are the coefficients representing the weights assigned to each feature X_i.
- n is the number of features.

SVM employs a kernel trick to transform the input space into a higher-dimensional space, allowing it to handle non-linearly separable data effectively. Common kernels include linear, polynomial, and radial basis function (RBF). The choice

of kernel and hyperparameters, such as the regularization parameter C and kernel parameters (e.g., γ for RBF), significantly influences the model's performance.

2) K-Nearest Neighbors (KNN): K-Nearest Neighbors (KNN) is a simple, yet powerful, instance-based learning algorithm used for classification and regression tasks. The algorithm operates by identifying the k nearest training examples to a given test point and predicting the output based on the majority class (for classification) or the average value (for regression) of those neighbors. KNN is particularly useful for its intuitive nature and effectiveness in certain scenarios.

The KNN prediction for a given instance x can be represented mathematically as follows:

$$\hat{y} = \frac{1}{k} \sum_{i=1}^{k} y_i \tag{2}$$

Equation 2. KNN Prediction Function

Where:

- $\hat{y}$ is the predicted output for the input instance x.
- k is the number of nearest neighbors considered.
- y_i are the outputs of the k nearest neighbors.

The distance metric (commonly Euclidean distance) plays a crucial role in determining the neighbors. The choice of k is also essential, as a small value can lead to a model that is sensitive to noise, while a large value may smooth out class distinctions. KNN is non-parametric and does not require an explicit training phase, which allows it to adapt quickly to changes in the dataset.

3) Summary: Both SVM and KNN are valuable algorithms for machine learning tasks, each with distinct strengths and applications. SVM excels in high-dimensional spaces and is robust against overfitting, while KNN is easy to implement and interpret, making it a great baseline model. The choice between these algorithms depends on the specific requirements of the task and the characteristics of the data.

E. Training Procedure

The researchers used K-fold Cross-Validation and this technique involves dividing the training dataset into k subsets (or folds). The model is trained on k-1 folds while validating it on the remaining fold. This process is repeated k times, with each fold serving as the validation set once. The cross-validation scores obtained from each fold are averaged to produce a more accurate assessment of the model's performance. In this study, a 5-fold cross-validation strategy was utilized.

```
[98] from sklearn.model_selection import cross_val_score

     #5-fold cross-validation
     scores = cross_val_score(model, X_train, y_train, cv=5)

     print("Cross-validation scores:", scores)
     print("Mean cross-validation score:", scores.mean())

     Cross-validation scores: [0.99963577 0.99981788 0.99981788 0.99945365 0.99981788]
     Mean cross-validation score: 0.9997086140957931
```

Fig. 4. Mean & Cross Validation Scores

For the SVM model, key hyperparameters tuned included:
C (Regularization Parameter): A standard value of 1 was used to avoid overfitting or underfitting.
Kernel Type: RBF and Linear were used to determine the most effective kernel.
Gamma: The researchers used the value, *scale*.
For the KNN model, the hyperparameter tuned was:
k (Number of Neighbors): Various values of k were tested to find the optimal balance between bias and variance.

F. Evaluation Metrics

The researchers used Accuracy, Classification Report, and Confusion Matrix as the model's evaluation metrics.

Accuracy: 84.8996 %
Classification Report:

	precision	recall	f1-score	support
0	0.92	1.00	0.96	331
1	1.00	0.99	1.00	432
2	0.95	1.00	0.97	310
3	0.96	1.00	0.98	245
4	0.95	0.99	0.97	498
5	0.82	0.87	0.84	247
6	0.93	0.94	0.94	348
7	1.00	0.95	0.98	436
8	0.81	0.88	0.85	288
9	0.82	0.63	0.71	331
10	0.89	1.00	0.94	209
11	0.82	0.78	0.80	394
12	0.90	0.66	0.76	291
13	0.92	0.93	0.93	246
14	1.00	1.00	1.00	347
15	0.99	1.00	1.00	164
16	0.26	0.47	0.33	144
17	0.70	0.74	0.72	246
18	0.80	0.70	0.75	248
19	0.56	0.67	0.61	266
20	0.82	0.60	0.69	346
21	0.62	0.76	0.68	206
22	0.78	0.75	0.76	267
23	0.86	0.76	0.81	332
accuracy			0.85	7172
macro avg	0.84	0.84	0.83	7172
weighted avg	0.86	0.85	0.85	7172

Fig. 5. Accuracy & Classification Report

Accuracy was used to provide a quick overview of the model's performance. Classification report on the other hand provides a much more in-depth view of the model's performance. Confusion Matrix makes it a visual representation of what labels were misclassified more often than the other labels.

G. Baselines and Comparative Models

KNN and Linear SVM was used as the baseline and comparative models in this study.
For KNN, an accuracy of 81% was observed.

KNN Model Evaluation
Accuracy: 0.8103736754043502
Classification Report:

	precision	recall	f1-score	support
0	0.83	1.00	0.91	331
1	0.94	0.94	0.94	432
2	0.97	1.00	0.98	310
3	0.77	0.93	0.84	245
4	0.77	0.97	0.86	498
5	0.98	0.95	0.96	247
6	0.91	0.94	0.92	348
7	0.94	0.97	0.95	436
8	0.89	0.67	0.76	288
9	0.88	0.58	0.70	331
10	0.96	0.90	0.93	209
11	0.76	0.52	0.62	394
12	0.77	0.60	0.68	291
13	1.00	0.91	0.95	246
14	1.00	1.00	1.00	347
15	0.95	1.00	0.97	164
16	0.33	0.62	0.43	144
17	0.65	0.83	0.73	246
18	0.72	0.70	0.71	248
19	0.45	0.69	0.54	266
20	0.69	0.58	0.63	346
21	0.67	0.61	0.64	206
22	0.80	0.69	0.74	267
23	0.95	0.68	0.79	332
accuracy			0.81	7172
macro avg	0.82	0.80	0.80	7172
weighted avg	0.83	0.81	0.81	7172

Fig. 6. KNN Accuracy & Classification Report

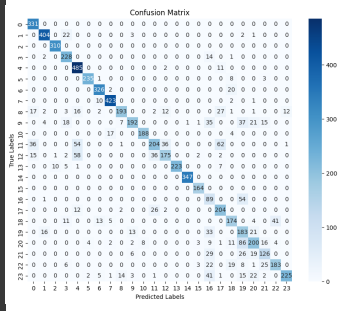

Fig. 7. KNN Confusion Matrix

For Linear SVM, an accuracy of 80% was observed.

Linear SVM Model Evaluation
Accuracy: 0.8053541550474066
Classification Report:

	precision	recall	f1-score	support
0	0.81	1.00	0.89	331
1	1.00	0.95	0.97	432
2	0.81	1.00	0.89	310
3	0.96	1.00	0.98	245
4	0.95	1.00	0.97	498
5	0.67	0.87	0.76	247
6	0.87	0.93	0.90	348
7	0.98	0.92	0.95	436
8	0.72	0.76	0.74	288
9	0.78	0.50	0.61	331
10	0.61	0.65	0.63	209
11	0.74	0.79	0.76	394
12	0.82	0.64	0.72	291
13	0.89	0.70	0.78	246
14	0.92	1.00	0.96	347
15	0.93	1.00	0.96	164
16	0.35	0.72	0.47	144
17	0.62	0.66	0.64	246
18	0.84	0.68	0.75	248
19	0.61	0.63	0.62	266
20	0.78	0.70	0.74	206
21	0.84	0.67	0.75	346
22	0.78	0.70	0.74	206
23	0.56	0.53	0.54	267
24	0.86	0.73	0.79	332
accuracy			0.81	7172
macro avg	0.79	0.79	0.78	7172
weighted avg	0.82	0.81	0.81	7172

Fig. 8. Linear SVM Accuracy & Classification Report

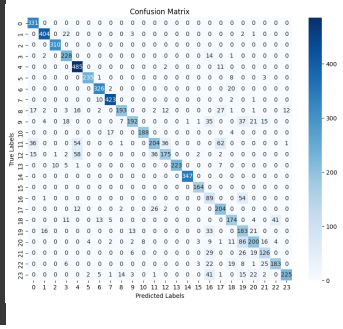

Fig. 9. Linear SVM Confusion Matrix

For RBF SVM, an accuracy of 84% was observed.

Accuracy: 84.8996 %
Classification Report:

	precision	recall	f1-score	support
0	0.92	1.00	0.96	331
1	1.00	0.99	1.00	432
2	0.95	1.00	0.97	310
3	0.96	1.00	0.98	245
4	0.95	0.99	0.97	498
5	0.82	0.87	0.84	247
6	0.93	0.94	0.94	348
7	1.00	0.95	0.98	436
8	0.81	0.88	0.85	288
9	0.82	0.63	0.71	331
10	0.89	1.00	0.94	209
11	0.82	0.78	0.80	394
12	0.90	0.66	0.76	291
13	0.92	0.93	0.93	246
14	1.00	1.00	1.00	347
15	0.99	1.00	1.00	164
16	0.26	0.47	0.33	144
17	0.70	0.74	0.72	246
18	0.80	0.70	0.75	248
19	0.56	0.67	0.61	266
20	0.82	0.60	0.69	346
21	0.62	0.76	0.68	206
22	0.78	0.75	0.76	267
23	0.86	0.76	0.81	332
accuracy			0.85	7172
macro avg	0.84	0.84	0.83	7172
weighted avg	0.86	0.85	0.85	7172

Fig. 10. RBF SVM Accuracy & Classification Report

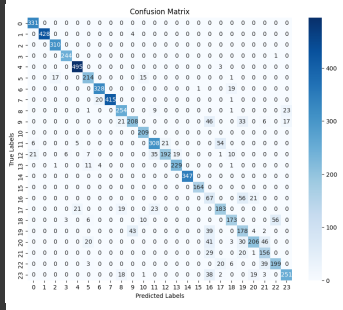

Fig. 11. RBF SVM Confusion Matrix

IV. Results And Discussion

A. Key Findings

Using the ASL Sign Language MNIST dataset, the researchers were able to produce a SVM Model with 84% accuracy. The classification of 24 sign languages, excluding *J* and *Z* due to gestural motion. KNN on the other hand had 81% and Linear SVM had 80%. These models were roughly split 80-20 train-test, with cross-validation to ensure robust performance.

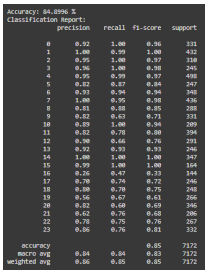

Fig. 12. RBF SVM Accuracy & Classification Report

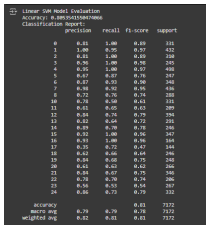

Fig. 13. Linear SVM Accuracy & Classification Report

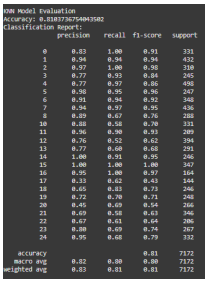

Fig. 14. KNN Accuracy & Classification Report

Although the RBF SVM performed better than KNN and Linear SVM, the results are not applicable for practical use. 84% accuracy will not be reliable when it comes to an even larger amount of dataset. A graph below compares the predicted and actual labels of the dataset.

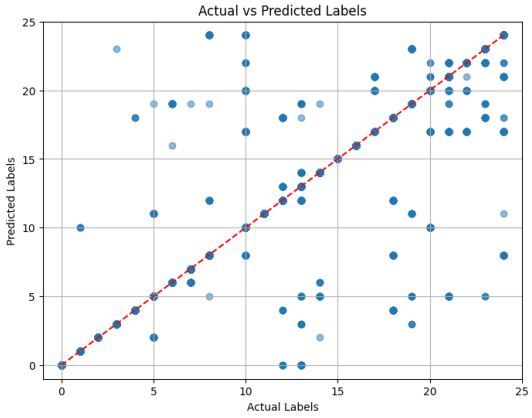

Fig. 15. Actual vs. Predicted Labels

V. Conclusion

The results were consistent with our expectations as the dataset were already outdated and had 28x28 pixels, which are very small compared to the latest images and technologies that can produce high quality images. Another factor is that while SVM is a good basic machine learning algorithm, it has it's limitations especially in image classification. Advanced machine learning algorithms such as Convoluted Neural Networks and Ensemble Learning will perform with a higher accuracy compared to SVM.

Some unanswered questions include how well would the model predict with images of higher quality and a larger image size. What are other ways of transforming the data to cater and help SVM produce a higher accuracy result.

One of the limitations of the study was the complex replicability of the image pipeline that the dataset was trained on, although, it may be because of the researcher's inept knowledge on the programming language used on the image pipeline. The image pipeline is vital in replicating the image that is in the training set.

In conclusion, this study shows that using SVM for ASL Recognition can be a powerful and precise model in classifying images. The study emphasizes the significance of proper hyperparameter setting and data pre-processing in achieving a successful and highly accurate model. This study adds to our understanding in image classification especially with American Sign Language (ASL).

REFERENCES

[1] Priyanka Bhandari, Ashish K. Yadav, Bhandari, P., Yadav, A. K., & Yadav, S. (2022). American Sign Language Recognition Using Support Vector Machine and Convolutional Neural Network. Artificial Intelligence Review, 55(5), 3521-3540. https://link.springer.com/article/10.1007/s10462-021-09996-w

[2] Madhav K. Sukhadiya, Mahesh N. Bansal, Tushar P. Parekh, (2019). Comparing ANN, SVM, and HMM Based Machine Learning Methods for American Sign Language Recognition Using Wearable Motion Sensors. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1245-1250). IEEE. https://ieeexplore.ieee.org/document/8666491.

[3] Khushbu Kachhadiya, Shital Kachhadiya, Ramesh Patel,(2021). American Sign Language Recognition Based on Machine Learning and Neural Networks. In 2021 2nd International Conference on Smart Technologies for Energy and Power (ICSTEP) (pp. 1-5). https://ieeexplore.ieee.org/document/9498024/authorsauthors

[4] Easa Alalwany, Moataz Al-Yahya, Mohamad Ali Al-Rababah,(2023). Deep Learning Technology to Recognize American Sign Language Alphabet. Sensors, 23(18), 7970. https://www.mdpi.com/1424-8220/23/18/7970

[5] Anjali P. K, Mohamad Ali Al-Rababah, Akhilesh Choudhary, (2021). Sign Language Recognition Based on Computer Vision. In 2021 International Conference on Intelligent Systems and Computing (pp. 1-5). https://ieeexplore.ieee.org/document/9498024

[6] Kumar, R. Anand, T. O. I. R. Prasath, 2020). Real-Time Sign Language Recognition. In 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1425-1430) https://ieeexplore.ieee.org/document/9220241

[7] https://www.kaggle.com/code/madz2000/cnn-using-keras-100-accuracy/notebookAnalysis-after-Model-Training

[8] https://www.kaggle.com/code/madz2000/cnn-using-keras-100-accuracy/notebookAnalysis-after-Model-Training

[9] https://ieeexplore.ieee.org/document/9622242

[10] https://ieeexplore.ieee.org/document/10670380